(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
20 December 2001 (20.12.2001)

PCT

(10) International Publication Number
WO 01/97010 A2

(51) International Patent Classification[7]: G06F 9/00

(21) International Application Number: PCT/EP01/05560

(22) International Filing Date: 14 May 2001 (14.05.2001)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
09/592,060 12 June 2000 (12.06.2000) US

(71) Applicant: KONINKLIJKE PHILIPS ELECTRONICS N.V. [NL/NL]; Groenewoudseweg 1, NL-5621 BA Eindhoven (NL).

(72) Inventor: RABELER, Thorwald; Prof. Holstlaan 6, NL-5656 AA Eindhoven (NL).

(74) Agent: PETERS, Carl, H.; Internationaal Octrooibureau B.V., Prof. Holstlaan 6, NL-5656 AA Eindhoven (NL).

(81) Designated State *(national)*: JP.

(84) Designated States *(regional)*: European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR).

Published:
— *without international search report and to be republished upon receipt of that report*

For two-letter codes and other abbreviations: refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

WO 01/97010 A2

(54) Title: DATA PROCESSING METHOD AND DEVICE FOR PROTECTED EXECUTION OF INSTRUCTIONS

(57) Abstract: A device to determine whether an application program has been unduly manipulated is provided with a signature register that is coupled to the output of the instruction register and/or to the addresses of the program memory. The signature register is also coupled to a mode bit of a processor having two operating modes. In a first mode, the signature register accumulates the output of the instruction register and/or the addresses of the program memory, yet the output of the signature register is not externally discernable. In a second mode, the signature register is set to no longer accumulate inputs. In the second mode, the signature register content is checked by a comparison with a specified value. This comparison value can be effectively contained in an instruction at relevant positions in the program. In one embodiment, the signature value is not compared directly but is only compared after the value is encrypted to ensure that in the case of unauthorized manipulation of the program, the comparison value cannot be readily modified.

Data processing method and device for protected execution of instructions

This invention generally relates to a method and data processing device for the execution of instruction sequences. Specifically, the present invention pertains to a method and data processing device for ensuring that a program portion has not been altered and is running in the proper sequence.

In program execution, program portions made up of instruction sequences have an intended sequence flow. Even in the case of an interrupt request, they are typically predefined interrupt exit points, interrupt entry points, and interrupt processing sequences. However, a problem may occur in the program sequencing. Errors may occur during the execution of series of instruction sequences, for example, due to electric interference signals or faulty components. Due to these errors, instructions may no longer be executed in the specified and intended order and the further execution of the program sequence may be permanently disturbed.

However, there are also cases where in proper operation, the instructions of a program are not executed in the specified order. These cases concern, for example, so-called smart cards. A smart card is a flat card that contains a data processing device in the form of a microprocessor as well as a memory. The smart card is operated in conjunction with a host device (e.g., a read apparatus) utilizing contacts or a wireless contact medium.

Smart cards of this kind may be utilized for banking applications where data in the smart card memory may be read out and/or modified only under stringently defined conditions. In addition, typically given other data (e.g., security-related data) is only exchanged between the smart card and the read apparatus and may not be externally disseminated or discerned. Such security-related data may also be involved in other applications of smart cards, for example their use as a health care card, a set top box access card, etc.

Typically, the program present in these smart cards is often modified or supplemented in order to add data to the smart card, to adapt the smart card to various applications, and/or to create further possibilities for use. Preferably, only the originator of

the smart card should realize these modifications or supplements. However, in practice it can hardly be precluded that unauthorized persons may attempt to modify the instruction sequences present in the microprocessor or in the program memory thereof, in order to carry out unauthorized reading or fraudulent modification of data.

In a prior art system described in German Patent Application No. 19804784.3, incorporated herein by reference, a smart card is proposed that offers a high degree of protection against this type of manipulation. To this end, the program portion that controls the smart card and reading apparatus interaction is subdivided into essentially two instruction sequences, with one instruction sequence being stored so that it cannot be modified. All accesses to essential, notably security-relevant data, are only possible in one instruction sequence that is referred to as the system program. The other instruction sequence is referred to as the application program. Both instruction sequences and programs may consist of a respective number of individual instruction sequences. When given, notably security-relevant data is to be accessed in the application program, a jump is made to the system program. Thereafter, a jump is made back to the application program and the execution thereof is continued. However, a problem exists in this system in that unauthorized modifications may be made in the unprotected application program. In doing this, intended entry and exit portions may be circumnavigated for the purposes of entering the system program in such a way as to either gain external access to security-relevant data or to cause the system program to run in an unauthorized manner.

In another prior art system, shown in U.S. Patent Application Serial No. 09/365,212, assigned to the same assignee of the present invention and incorporated herein by reference, a signature register is utilized to accumulate a sequence of addresses or instructions. The accumulated addresses or instructions from the signature register are compared to a value contained in an instruction portion to determine if the intended instruction flow has been improperly modified. In a case wherein an improper instruction flow has occurred, the value stored in the signature register will not correspond to the value in the instruction portion and operation of the device is terminated. To foil an unauthorized modification of the value in the signature register, an encryption device is utilized at times to modify the value in the signature register in a secret way. However, if the method of encryption in the encryption device is discovered through the use of unauthorized tampering, the security of the device may be compromised.

Accordingly, it is an object of the present invention to overcome the disadvantages of the prior art.

It is a further object of the present invention to provide a method and a data processing device wherein an unauthorized modification or supplementation of instructions may be recognized.

A device to determine whether an application program has been unduly manipulated is provided with a signature register that is coupled to the output of the instruction register and/or to the addresses of the program memory. The signature register is also coupled to a mode bit of a processor having two operating modes. In a first mode, the signature register accumulates the output of the instruction register and/or the addresses of the program memory, yet the output of the signature register is not externally discernable. In a second mode, the signature register is set to no longer accumulate inputs. In the second mode, the signature register content is checked by a comparison with a specified value. This comparison value can be effectively contained in an instruction at relevant positions in the program or may be stored in an unalterable memory location. In one embodiment, the signature value is not compared directly but is only compared after the value is encrypted to ensure that in the case of unauthorized manipulation of the program, the comparison value cannot be readily modified.

In one embodiment, the signature register is coupled to the addresses of the program memory through a memory management unit. In this way, the signature register may accumulate virtual addresses of the program memory in place of the actual addresses of the program memory thereby facilitating characterization of the program flow. The virtual addresses may correspond to self-relative addresses.

In an embodiment, the first mode of the processor is reserved for untrusted application program sequences while the second mode of the program is reserved for trusted system program sequences. Untrusted program sequences are program sequences that are changeable and discernable by intervention outside the device. Trusted program sequences are program sequences that are unchangeable and not discernable by intervention outside the device. In particular embodiments the device may correspond to a smart card.

The following are descriptions of embodiments of the present invention that when taken in conjunction with the following drawings will demonstrate the above noted features and advantages, as well as further ones. It should be expressly understood that the drawings are included for illustrative purposes and do not represent the scope of the present

invention. The invention is best understood in conjunction with the accompanying drawing in which:

Fig. 1 is a block diagram of a data processing device in accordance with an embodiment of the present invention; and

Fig. 2 shows a flow chart for the execution of instruction sequences in accordance with an embodiment of the present invention.

In the discussion to follow, certain terms will be illustratively discussed in regard to specific embodiments or systems to facilitate the discussion. However, as would be readily apparent to a person of ordinary skill in the art, these terms should be understood to encompass other similar applications and embodiments wherein the present invention could be readily applied.

Fig. 1 shows a block diagram of a data processing device 100 in accordance with the invention. A memory 10 contains sequences of instructions that are addressed by an address generator 12 via a lead 13A and thereby read out successively. The instructions read out are applied to an instruction register 14 which stores each time at least one instruction. The instructions or at least parts thereof, are applied to various elements via the lead 15.

One of these elements is an instruction decoder 16. The instruction decoder 16 notably decodes the operation part of the instruction and conducts signals, via corresponding control leads, to other elements that are in this case shown in general as an execution device 18. The execution device 18 includes notably a processor, such as a central processing unit ("CPU") as well as registers (not shown).

A lead 19 from the execution device 18 to the address generator 12 conducts control signals to the address generator 12 when the execution of an instruction is to be terminated and the next instruction is to be called. Additionally, the lead 19 conducts control signals to the address generator 12, for example, in response to an instantaneous instruction (e.g., an interrupt instruction) or when on a state of the execution device 18, a different instruction portion is requested from the address generator 12. The new address of the different instruction portion is preferably indicated, via the lead 15, by a part of the instruction present in the instruction register 14. In this case, this new address together with the control signal on lead 19 triggers a jump in the instruction sequence.

In one embodiment, the lead 15 is also connected to an input of a signature register 20. The signature register 20 is also connected, via a lead 13B, to the output of the address generator 12. In other embodiments it is also possible for only one of these two leads 13B and 15, to be connected to the signature register 20. The signature register 20 accumulates newly received values from either or both of the address generator 12 and the instruction register 14 with the value stored in the signature register 20. The signature register 20 stores the accumulated value in response to a received relevant signal from the instruction decoder 16 via a lead 17A. It should be clear that the signature register 20 may accumulate all received values or any portion thereof to confound any intentional malicious intervention.

The signature register 20 may accumulate values in many known ways such as maintaining a checksum of the accumulated data, an exclusive-or (XOR) of accumulated data, or other known accumulation methods. In addition, individual bits of incoming values may be combined/accumulated and thereafter may be accumulated to stored accumulation result. Significantly, although the accumulation methods may be known in the art, the particular method selected may be unknown to further foil attempts at concealing an unauthorized modification or supplementation of instructions. In one embodiment, all bits of incoming values may be accumulated wherein in other embodiments, only selected bits may be accumulated or even some combination thereof to further provide unpredictability (apparent random behavior) of the accumulated data.

The output 21 of the signature register 20 is illustratively connected to an input of a comparator 22. A second input of the comparator 22 is connected to the lead 15 in the illustrative embodiment. In this way, the comparator 22 compares the value output by the signature register 20 via the lead 21 with a value contained in an instruction portion when the instruction decoder 16, via the lead 17B, supplies a corresponding signal. In other embodiments, the comparator 22 may compare the value output by the signature register 20 with a value contained in an unalterable portion of the memory 10 via the instruction register 14 and lead 15.

In a case wherein the value output by the signature register 20 via the lead 21 does not correspond with a value contained in an instruction portion, the comparator 22 outputs, via the lead 23, a value and a control signal to the address generator 12. In response to the control signal, the address generator 12 is set to a given address so that a predetermined instruction sequence is executed. The predetermined instruction sequence may, for example, be an interrupt instruction sequence that inhibits further functions of the device.

In one embodiment, the signature register 20 may also contain an encryption device 24 that generates a new value from the combined value stored in the signature register 20 using a secret algorithm. This new value is stored in the signature register 20 in the presence of a relevant control signal on the lead 17A. The signature register 20 then commences the further signature formation on the basis of a value that cannot be predicted by a person not knowing the encryption algorithm in the encryption device 24. Consequently, it is rendered difficult to modify the value applied to the comparator 22 by a relevant instruction in such a manner that in the case of a modification of the previous instruction sequence, the comparator 23 does not produce a signal or perform a jump into an interrupt instruction sequence. In this way, it can be checked whether the instruction sequences and the instructions contained therein are executed in the correct order and whether instructions therein have been modified.

The additional encryption and protection provided by the encryption device 24 may further be utilized when the checksum is communicated to the outside, such as through a card reader 28. Encrypting the checksum output by the signature register 20 makes it more difficult for a hacker to simply tap a lead (e.g., a telephone line) of a corresponding device, such as an automatic teller machine (ATM), and patch in a good checksum.

Further, in accordance with the present invention, a mode signal output from the execution device 18 on lead 17C acts as a hardware protection that excludes the application mode from accessing and/or modifying the signature register 20. In this way, the device 100, such as a smart card, has distinct operating modes as determined by a hardware constraint in the form of the mode signal.

In one embodiment, the "application mode" or "user mode" is determined by a mode signal logical state of zero ("zero mode signal"). The zero mode signal is reserved for executing non-trusted applications. A mode signal logical state of one ("one mode signal") is reserved for the system program, including the execution of interrupt code portions. In this way, the mode signal is part of the processor state (e.g., the CPU of the execution device 18). Accordingly, an application program is excluded by the hardware constraint (e.g., the mode signal), from increasing the operating mode to "system mode", other than by passing full control to operating system code thru predetermined entry points.

In one embodiment, the application program invokes "system call" instructions. The system call instructions from the instruction register 14 triggers an interrupt instruction portion. In this embodiment, the operating system code acts as an interrupt handler for these "system call" interrupts.

In operation, the signature register 20 may build up a checksum or other logical operations of executed instructions and/or instruction addresses, while the CPU is running in the application mode. The signature register 20 is not accessible in any way by the application program since the zero mode signal acts as a hardware lock that is set in the application mode. The zero mode signal thus prevents access, by the application program, to the output 21 of the signature register 20.

In the system mode, as hardware determined by the one mode signal, the checksum state of the signature register 20 is unaffected by additional address generation and is accessible for read operations. Since all interrupt code is executed in the system mode, the interrupt code is also excluded from being summed by the signature register 20. In addition, the mode of an interrupted program is saved/restored on interrupt entry/exit, so the signature register 20 resumes checksumming only once control is returned to the application mode by an interrupt exit. The signature register 20 will not resume checksumming if control is returned to a previously interrupted system program. In this way, the mode of the CPU controls the accumulate/stop state and access to the result (e.g., checksum) of the signature register 20.

The exclusion of the interrupt code from the result of the signature register 20 is desirable to facilitate characterization of the application program and its flow by a unique signature. Inclusion of the interrupt code portions in the result renders the result dependent on the exact time or instruction portion where the interrupt code was executed with respect to the application program. This makes it more difficult to properly characterize the instruction flow by the result contained in the signature register 20 since the occurrence of an interrupt is not restricted to a given particular portion of the application program.

In the system mode, the system program has full control over the result of the signature register 20. There is no need to protect the signature register 20 from alteration by the system program, since the system program is "TRUSTED" code, that is not modifiable from the "NONTRUSTED" application program, as controlled by a hardware protection mechanism as described in the German Patent Application No. 19804784.3. As shown therein, the system program is also in full control of the memory protection hardware, so the system program can protect itself from access by the application program.

In one embodiment, the address generator 12 acts as a memory management unit that performs relocation of the addresses for accessing the memory 10 and for controlling addresses that may be output to the signature register 20 via the lead 13B. This is desirable since the signature register 20 may, in one embodiment, perform a checksum of the

addresses, yet application programs may be loaded on the fly without defined (e.g., fixed) memory locations.

Without the relocation of the address performed by the memory management unit, the addresses, such as those referenced within the program (e.g., JUMP to address XX) should be changed for the final location of the program in memory. A relocating loader may perform this change in the addresses. However, then the checksum in the signature register may no longer characterize the program, but the program at its new address. The relocating loader is a utility program that typically modifies the address portion of a program, while it is brought into memory, e.g., before execution. The relocation loader prior to program execution goes through the program and changes all references from a symbolic address (e.g., an address relative to the program flow and not relative to the actual location in memory where the program is stored) to the actual addresses where the program is stored. The problem with this is that the symbolic addresses are typically lost and therefore, may not be utilized by the signature register to characterize the program flow.

Utilizing the address generator 12 as a memory management unit may solve this problem. With the memory management unit, the program flow may at times refer to virtual addresses that are unchanged prior to execution. The virtual addresses are output by the address generator 12 to the signature register 20 via the lead 13B so that the checksum in the signature register is unaffected by the actual addresses where the program is located in the memory 10. The memory management unit performs the change of virtual addresses to the actual addresses and outputs these actual addresses via the lead 13A to the memory 10.

In this embodiment, the memory management unit or other means of address independence is desirable to facilitate the characterization of the program by the programs address sequence. The memory management unit is under full control of the system program. The system mode program "knows" the correspondence of application program virtual address to actual address. In this way, the application program only refers to the virtual addresses, which are unchanged, regardless of the location of the application program in memory. In this way, the signature register may characterize the program flow (by the virtual addresses) without being affected by the actual addresses.

In an alternate embodiment, the program portion may utilize self-relative addressing. Self-relative addressing renders the program portion "position-independent". This approach utilizes addressing modes in the CPU, like relative jumps, to ensure that the checksum in the signature register 20 characterizes the program flow. In self relative addressing, the application program addresses are relative to some basis address, e.g.,

reflecting only the distance of a branch target from the current instruction. As long as the application program is moved in one piece, all branches inside the application program are unchanged, as the relative location of start and end of a branch move together.

In one embodiment, the device 100 shown in Fig. 1 constitutes the control device of a smart card as described in the cited German patent application 19804784.3. In this embodiment, the control signal on the lead 17b may activate the comparator 22 and is generated preferably for each jump instruction whereby a jump to the described system program is executed. At the same time, the signature register 20 is set to not accumulate by a change in the mode signal from the CPU. The system program is preferably stored in a part of the memory 10 that cannot be modified as discussed above. The signature register 20 is released to accumulate again by the mode signal only upon a return jump to the application program. In one embodiment, the value generated by the encryption unit 24 may then be stored as the checksum in the signature register 20 in response to a signal on a lead 29 from the smart card or on the lead 17A from the instruction decoder 16.

Typically, smart cards operate in cooperation with a card reader. The block 28 in Fig. 1 symbolically represents such a card reader. When the program in the smart card performs a jump, the card reader 28 may receive, for example, the value at the output 21 of the signature register 20. The output 21 may represent the checksum directly or may represent the value encrypted by the encryption unit 24. This value can then be compared with a predetermined value in the card reader 28. In one embodiment, the comparator 22 may be omitted if the card reader 28 is capable of controlling the smart card accordingly.

In another embodiment, the card reader 28 may form a further signature from said received value and may return this further signature to the signature register 20 via the lead 29. The smart card may thus check whether it is cooperating with an appropriate, notably non-manipulated card reader.

Fig. 2 shows an illustrative flow chart of an execution of instruction sequences in accordance with an embodiment of the present invention. As shown, the execution commences with a start 30 that symbolizes the introduction of the smart card into the card reader thereby establishing a conductive connection or a contactless connection. Subsequently, a start program 31 is performed during which the card and the card reader exchange various data, for example in order to determine the nature of the card, whether authorization tests have to be performed, etc. Subsequently, the processing proceeds with a program sequence 32 that forms a portion of an application program. Prior to executing program sequence 32 (e.g., as a last part of program sequence 31 or prior thereto), the mode

signal of the CPU of the execution device 18 is cleared to zero thereby setting the signature register 20 to begin accumulating.

The program sequence 32 contains, in a predetermined position, a jump instruction to the system program. The CPU sets the mode bit, thereby setting the signature register 20 to stop accumulating. Thereafter, the processing proceeds with a sequence 33 of system instructions. The sequence 33 verifies the signature previously formed during the processing of the program sequence 32. Subsequently, further predetermined system instructions 34 are carried out, after which a return jump to the application program is made and the CPU clears the mode bit setting the signature register 20 to accumulate. The application program continues at sequence 35 during which new signatures are formed by the signature register 20. At a given position of the application program at sequence 35, or when a given condition is satisfied, again a jump is made to the system program and the CPU sets the mode bit, thereby setting the signature register 20 to stop accumulating. The processing then proceeds with system instructions at sequence 36 and the signature formed is again tested. Additional system instructions may continue at sequence 37.

In one embodiment, the sequence 37 may incorporate the generation of a new initial value, being an erasure or other alteration of the checksum stored in the signature register such as an encryption of the previously formed checksum. The altered checksum may be written to the signature register 20 by the system program during sequence 37 of the system program. In this case, the altered checksum may be utilized as the basis for further checksum generation upon a return to an application program sequence.

The foregoing may be repeated several times until ultimately the desired operation with the card has been executed by means of the instruction sequence 38 of the application program. Upon such completion, the program returns to the system program again by way of a jump instruction and the CPU sets the mode bit, thereby stopping the signature register 20 from accumulating. The signature at the output of the signature register 20 is thus checked by means of the system instructions 39. Thereafter, terminating instructions 40 of the system program are executed until the program is ultimately stopped and the card is ejected as denoted by an end sequence 42.

When the testing of the value from the signature register by way of the instructions at sequences 33, 36, or 39 reveals that an expected value is not present, a jump takes place to an interruption program at sequence 41. The interruption program inhibits all further external operations of the card and directly leads to the end 42.

Finally, the above-discussion is intended to be merely illustrative of the invention. For example, the system program may dispatch control to several application programs in a controlled fashion (e.g. in a cyclic mode for timesharing). At the same time, the system program may save and restore the corresponding intermediate results from/to the signature register 20, such that each separate application program execution only updates the result of the corresponding application program.

As an example, the system program may load a checksum A into the signature register 20, execute instructions of an application program A and accumulate to the checksum A, then save the accumulated checksum A. Thereafter, the system program may load a checksum B into the signature register 20, execute part of an application program B and accumulate to the checksum B, and then save the checksum B. The system program may then restore the checksum A into the signature register 20 and resume execution of the application program A, and so on. After both (or more) application programs are executed to some known point, the cumulative checksum of each of the application programs may be evaluated. This enables "timesharing" of not only the processor for several application programs at quasi the same time, but also "timesharing" of the checksum mechanism, such that the signature register 20 operates independently for all of the application programs involved at quasi the same time.

As another example, although FIG. 1 shows separate functional blocks for the identified functions, one or more of these functions may be combined into a single block or divided into separate functional blocks. Clearly any one or more of these functional blocks may be performed via an instruction sequence of a processor, may be performed by a hard-wired integrated circuit, may be performed by a re-programmable integrated circuit, or any other known means or combination thereof.

Numerous alternative embodiments may be devised by those having ordinary skill in the art without departing from the spirit and scope of the following claims. In interpreting the appended claims, it should be understood that:

a) the word "comprising" does not exclude the presence of other elements or acts than those listed in a given claim;

b) the word "a" or "an" preceding an element does not exclude the presence of a plurality of such elements;

c) any reference signs in the claims do not limit their scope; and

d) several "means" may be represented by the same item or hardware or software implemented structure or function.

CLAIMS:

1. A method of executing a sequence of program instructions in a processor comprising:

- accumulating data from a first portion of the sequence of program instructions without providing an indication of the current value of the accumulated data in response to a first mode signal from the processor; and

- determining the accumulated data from the first portion of the sequence of program instructions during a second portion of the sequence of program instructions in response to a second mode signal from the processor.

2. The method of Claim 1, comprising comparing the determined accumulated data from the first portion of the sequence of program instructions to a value and further executing the program in dependence on the comparison.

3. The method of Claim 1, wherein the data accumulated from the first portion of the sequence of program instructions is at least one of the program instructions, corresponding actual memory address locations for the program instructions, and corresponding virtual memory address locations for the program instructions.

4. The method of Claim 1, wherein the first portion of the sequence of program instructions is a portion of an unprotected application program and the second portion of the sequence of program instructions is a portion of a protected system program.

5. The method of Claim 1, comprising changing corresponding address locations from actual addresses of the first portion of the sequence of program instructions in memory to virtual addresses wherein the data accumulated from the first portion of the sequence of program instructions are the virtual addresses for the program instructions.

6. The method of Claim 5, wherein the virtual addresses are self-relative addresses.

7. The method of Claim 2, comprising modifying the data accumulated from the first portion of the sequence of program instructions prior to comparing the determined accumulated data to the value.

8. The method of Claim 7, wherein the accumulated data is modified by encryption.

9. A processing device (100), comprising:

- a processor (16) configured to execute a first instruction sequence and output a corresponding first mode signal and to execute a second instruction sequence and output a corresponding second mode signal; and
- a signature register (20) configured if receiving the first mode signal to accumulate data corresponding to the first instruction sequence and provide no indication of the accumulated data, and configured if receiving the second mode signal to stop accumulating data and to provide an indication of the accumulated data.

10. The processing device (100) of Claim 9, comprising a comparator (22) coupled to an output of the signature register (20) and configured to receive the accumulated data and compare the accumulated data to a value, wherein the processor (16) is configured to execute an additional instruction sequence in dependence on the result of the comparison.

11. The processing device (100) of Claim 9, the device comprising a memory manager (12) coupled to the signature register (20) and configured to change corresponding address locations of the first instruction sequence to virtual addresses, and wherein the signature register is configured to accumulate the virtual addresses.

12. The processing device (100) of Claim 11, wherein the virtual addresses correspond to self-relative addresses.

13. The processing device (100) of Claim 9, wherein the processing device (100) is a portion of a smart card.

14. A program stored on a processor readable memory device having protected and unprotected memory portions, the program comprising:

- a first program sequence stored in the unprotected memory portion for execution and accumulation of a corresponding program portion during a first processor mode state; and

- a second program sequence stored in the protected memory portion for execution and comparison of the accumulated program portion during a second processor mode state.

15. The program of Claim 14, wherein the second program sequence contains a value that is compared to the accumulated portion and the program is configured to perform further execution in dependence on the results of the comparison.

16. The program of Claim 15, wherein the program is configured to compare an encryption of the accumulated portion to the value.

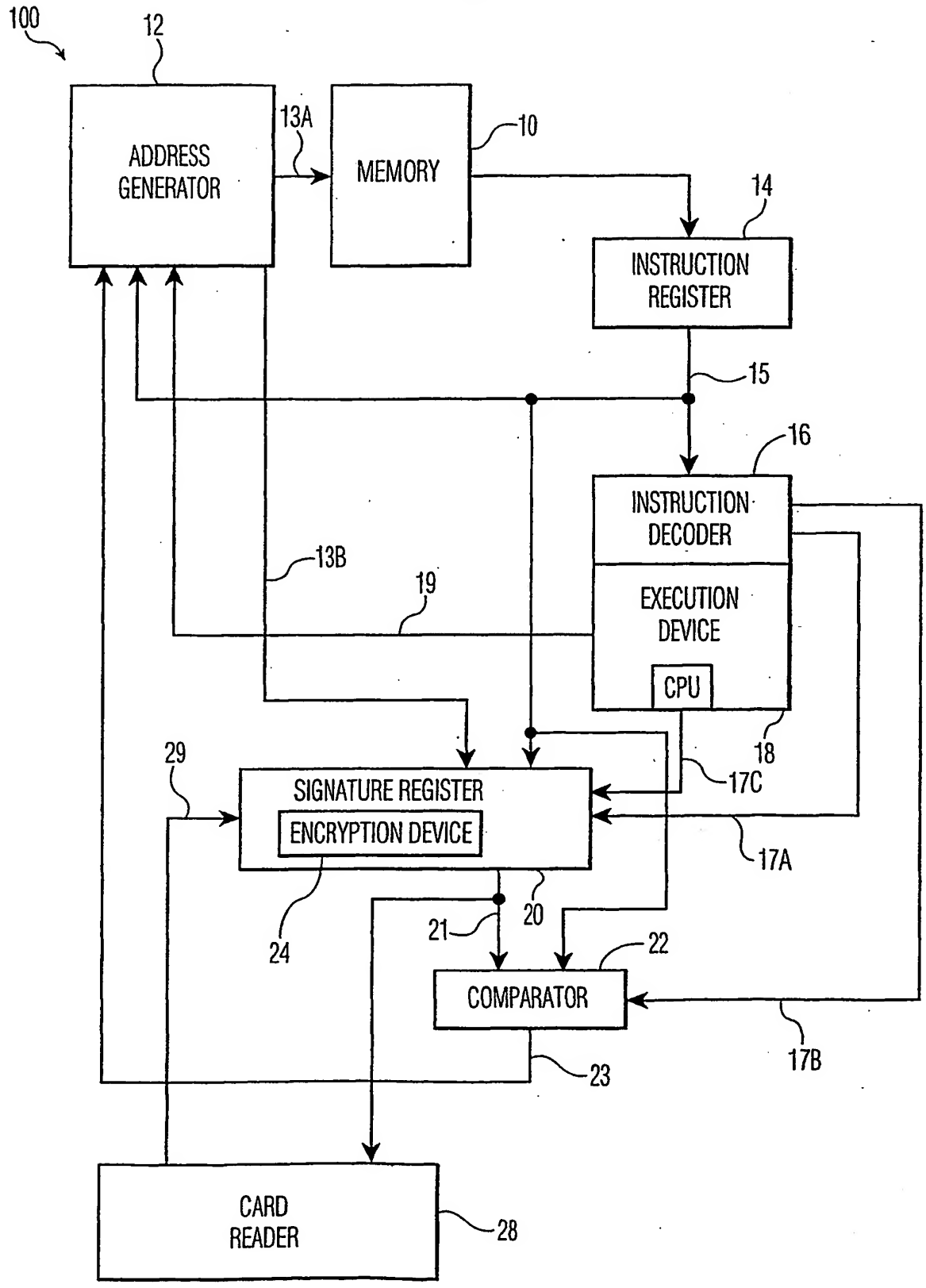


FIG. 1

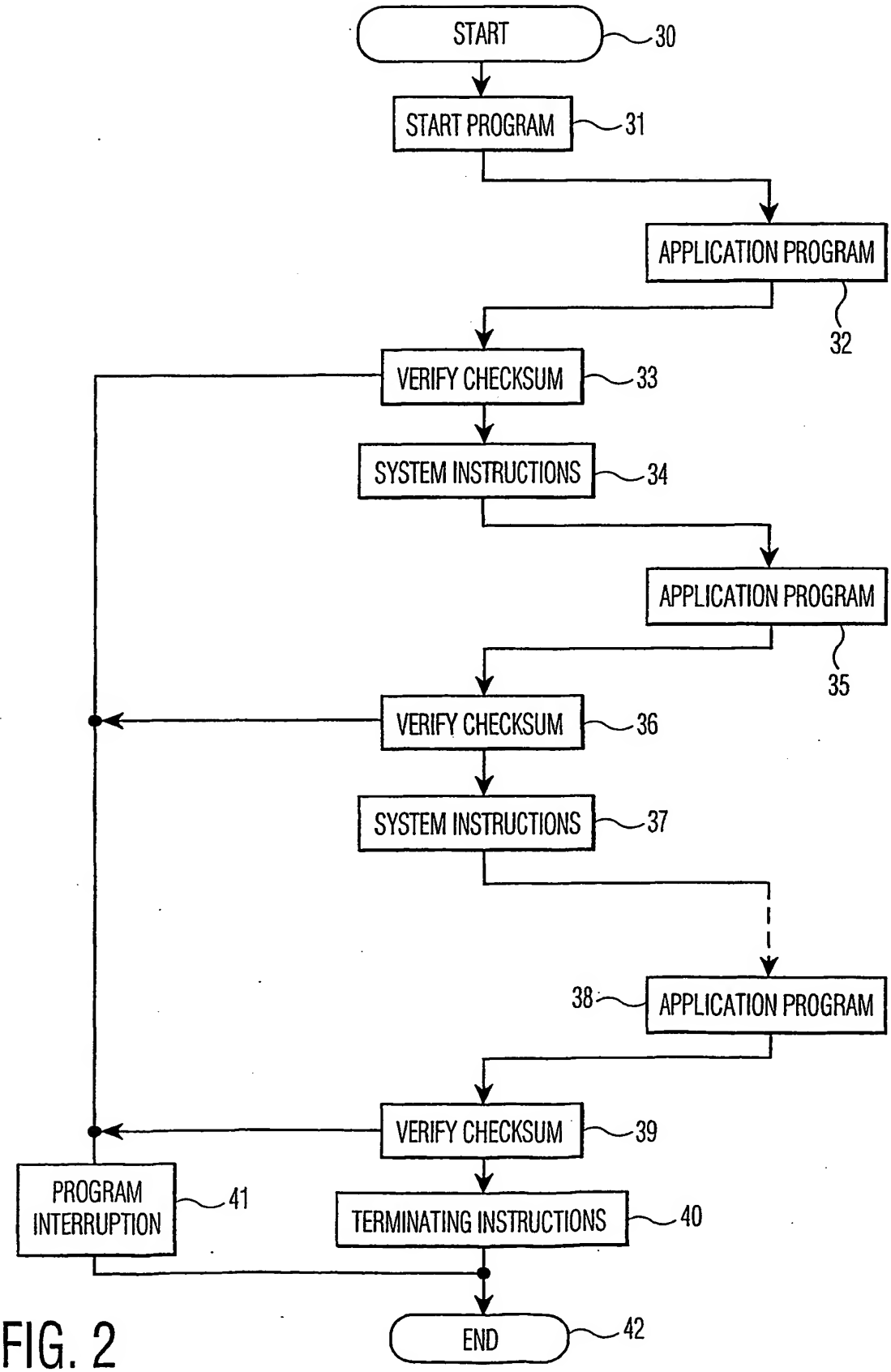

START
30
START PROGRAM
31
APPLICATION PROGRAM
32
VERIFY CHECKSUM
33
SYSTEM INSTRUCTIONS
34
APPLICATION PROGRAM
35
VERIFY CHECKSUM
36
SYSTEM INSTRUCTIONS
37
38
APPLICATION PROGRAM
VERIFY CHECKSUM
39
PROGRAM INTERRUPTION
41
TERMINATING INSTRUCTIONS
40
END
42

FIG. 2

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
20 December 2001 (20.12.2001)

PCT

(10) International Publication Number
WO 01/97010 A3

(51) International Patent Classification[7]: G06F 1/00, G07F 7/10, G06F 11/28

(21) International Application Number: PCT/EP01/05560

(22) International Filing Date: 14 May 2001 (14.05.2001)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
09/592,060 12 June 2000 (12.06.2000) US

(71) Applicant: KONINKLIJKE PHILIPS ELECTRONICS N.V. [NL/NL]; Groenewoudseweg 1, NL-5621 BA Eindhoven (NL).

(72) Inventor: RABELER, Thorwald; Prof. Holstlaan 6, NL-5656 AA Eindhoven (NL).

(74) Agent: PETERS, Carl, H.; Internationaal Octrooibureau B.V., Prof. Holstlaan 6, NL-5656 AA Eindhoven (NL).

(81) Designated State *(national)*: JP.

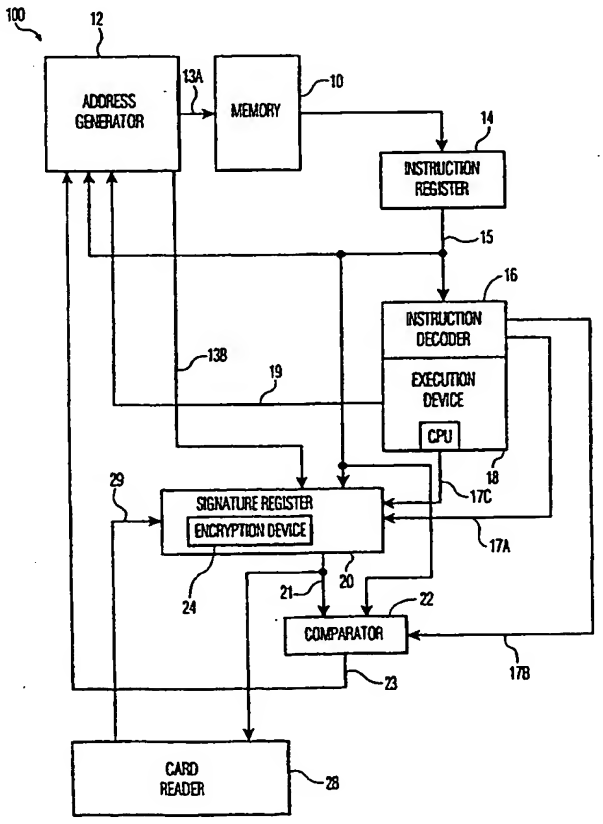
(84) Designated States *(regional)*: European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR).

Published:
— *with international search report*

(88) Date of publication of the international search report:
21 March 2002

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(54) Title: DATA PROCESSING METHOD AND DEVICE FOR PROTECTED EXECUTION OF INSTRUCTIONS



(57) Abstract: A device to determine whether an application program has been unduly manipulated is provided with a signature register that is coupled to the output of the instruction register and/or to the addresses of the program memory. The signature register is also coupled to a mode bit of a processor having two operating modes. In a first mode, the signature register accumulates the output of the instruction register and/or the addresses of the program memory, yet the output of the signature register is not externally discernable. In a second mode, the signature register is set to no longer accumulate inputs. In the second mode, the signature register content is checked by a comparison with a specified value. This comparison value can be effectively contained in an instruction at relevant positions in the program. In one embodiment, the signature value is not compared directly but is only compared after the value is encrypted to ensure that in the case of unauthorized manipulation of the program, the comparison value cannot be readily modified.

WO 01/97010 A3

INTERNATIONAL SEARCH REPORT

[redacted] national Application No
[redacted]T/EP 01/05560

A. CLASSIFICATION OF SUBJECT MATTER
IPC 7 G06F1/00 G07F7/10 G06F11/28

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 7 G07F G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)
EPO-Internal

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X Y	EP 0 977 160 A (PHILIPS CORP INTELLECTUAL PTY ;KONINKL PHILIPS ELECTRONICS NV (NL)) 2 February 2000 (2000-02-02) the whole document	1-4,7,8 9,10, 13-16
Y	EP 0 935 214 A (PHILIPS PATENTVERWALTUNG ;KONINKL PHILIPS ELECTRONICS NV (NL)) 11 August 1999 (1999-08-11) cited in the application column 1, line 28 -column 2, line 12	9,10, 13-16
A	US 5 671 394 A (KATSUTA HIROSHI) 23 September 1997 (1997-09-23) the whole document	1,9, 14-16

-/--

[X] Further documents are listed in the continuation of box C.

[X] Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
30 November 2001	06/12/2001

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Klocke, L

Form PCT/ISA/210 (second sheet) (July 1992)

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 6 003 134 A (LO MINWEN ET AL) 14 December 1999 (1999-12-14) the whole document	1,9,14

INTERNATIONAL SEARCH REPORT

[redacted]formation on patent family members

[redacted]national Application No
[redacted]T/EP 01/05560

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP 0977160 A	02-02-2000	DE 19834486 A1	03-02-2000
		EP 0977160 A1	02-02-2000
		JP 2000112749 A	21-04-2000
EP 0935214 A	11-08-1999	DE 19804784 A1	12-08-1999
		EP 0935214 A2	11-08-1999
		JP 11272828 A	08-10-1999
US 5671394 A	23-09-1997	JP 4084348 A	17-03-1992
		DE 69130750 D1	25-02-1999
		DE 69130750 T2	02-09-1999
		EP 0468535 A2	29-01-1992
US 6003134 A	14-12-1999	US 5754762 A	19-05-1998
		AU 5700698 A	03-08-1998
		WO 9830958 A1	16-07-1998